DOCUMENT RESUME

ED 220 036 HE 015 395

AUTHOR Stevenson, Michael; Walleri, R. Dan
TITLE Application Generators: Their Impact within Institutional Research and Beyond. AIR Forum 1982 Paper.
PUB DATE May 82
NOTE 28p.; Paper presented at the Annual Forum of the Association for Institutional Research (22nd, Denver, CO, May 16-19, 1982).

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS Case Studies; *College Administration; Computer Oriented Programs; *Computer Programs; Databases; Higher Education; *Information Needs; *Institutional Research; *Management Information Systems; Programing Languages; Technological Advancement
IDENTIFIERS *AIR Forum

ABSTRACT

The characteristics of application generators and the reasons they are and will be used increasingly in collegiate institutional research are addressed. The incorporation of application generators within a total institutional data and information processing network is also outlined. The analysis considers the often noted "gap" between expectations and results in the application of computer power to information needs. It is proposed that application generators will eventually displace COBOL and other third generation languages as the primary tool for information system development. It is further argued that institutional research represents the most logical candidate for acting as the catalyst in this transformation. Evidence in support of these arguments is derived from a review of relevant literature and from a detailed description of one application generator and its use at one institution. In addition, strategies for beginning the transition to application generators and related institutional issues are examined. Application generators are classified as fourth generation software since they are non-procedural. Third generation languages require the user to describe not only what is desired, but also to tell the computer how to do it. Application generators require only the description of what is desired. Other characteristics of application generators are considered, and examples of application generators are noted. The case of a college that converted to an application generator and used the application generator package "INFO" is described. INFO's significant impact for institutional research and other college applications are addressed. A bibliography is appended. (SW)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED220036

APPLICATION GENERATORS: Their Impact Within Institutional Research and Beyond

Dr. Michael Stevenson
Director of Research and Computer Education
Mt. Hood Community College
26000 SE Stark Street
Gresham, OR 97030 503-667-7298

Dr. R. Dan Walleri
Institutional Researcher
Mt. Hood Community College
26000 SE Stark Street
Gresham, OR 97030 503-667-7146

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Association for Institutional Research
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
This document has been reproduced as received from the person or organization originating it.
Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

Prepared for Presentation at the Twenty-Second Annual Forum of the Association for Institutional Research, Denver, Colorado, May 16-18, 1982.

HE 015 395

THE ASSOCIATION FOR INSTITUTIONAL RESEARCH

This paper was presented at the Twenty-Second Annual Forum of the Association for Institutional Research held at the Denver Hilton Hotel in Denver, Colorado, May 16-19, 1982. This paper was reviewed by the AIR Forum Publications Committee and was judged to be of high quality and of interest to others concerned with the research of higher education. It has therefore been selected to be included in the ERIC Collection of Forum papers.

D. R. Coleman, Chairman
Forum Publication
Advisory Committee

ABSTRACT

This paper addresses the often noted 'gap' between expectations and results in the application of computer power to information needs. This problem is explored within the framework of institutional research in particular and higher education management in general. It is argued that a 'hardwired' MIS system is not the solution. Rather, recent advances in computer software, in the form of application generator languages, offer the closest approximation and most cost effective means of having information at the decision-makers 'finger tips'.

INTRODUCTION

This paper describes the characteristics of application generators and why they are and will increasingly be used in institutional research. Further, the incorporation of application generators within a total institutional data and information processing network is outlined. The proposition developed is that application generators will eventually displace COBOL and other third generation languages as the primary tool for information system development. It is further argued that institutional research represents the most logical candidate for acting as the catalyst in this transformation. Evidence in support of these arguments is derived from a review of relevant literature and from a detailed description of one application generator and its use at one institution. Finally, strategies for beginning the transition to application generators and accompanying institutional issues are examined.

WHAT ARE APPLICATION GENERATORS

Context of Available Software

Application generators are fourth generation software languages. In order to understand the significance of application generators, it is important to review the history of computer software in terms of generations. Each change in software generation along with the corresponding changes in computer hardware has

significantly changed the view of what is economically possible from computers. The machine languages of the first software generation demonstrated that computers could be programmed to perform processing tasks. The assembly languages of software's second generation expanded the tasks which could be efficiently programmed. While the macro languages such as COBOL, FORTRAN, and BASIC of software's third generation did not improve on the efficiency of the assembly languages, they opened programming up to many users by making the coding task correspond to fairly simple English commands.

Application generators are classified as fourth generation software since they are non-procedural. As early as 1978, McCraken recognized the importance of non-procedural languages. Watkins (1980), Dalton (1981), Hoffman (1981), Schindler (1981); as well as Lerner, Brownstein and Smith (1982) describe the difference between third and forth generation software. The third generation languages require the user to describe not only what is desired, but also to tell the computer how to do it. Application generators require only the description of what is desired.

Application generator software has the purpose of giving the user the necessary tools (commands) to generate an information system. That is, to create a screen for inputting data; to define a file, it's records, and fields; and to create a report. Similarly other general purpose software packages place word processing and statistical analysis tools within the easy use of computer users.

Characteristics of Application Generators

The characteristics of application generators are:

1. they use non-procedural commands to create screens, files, and reports;
2. they use 'relational' data base techniques to access and maintain connections between data items;
3. they access external files;
4. they interface with third generation languages such as COBOL;
5. they can select subsets of created files; and
6. they usually contain a generalized report writer.

Bradley (1982) explains the simple, yet powerful technique of building 'relational' data bases. This technique efficiently allows for a non-redundant data base where the connections can be made to related data items. Further, an application generator's ability to access external files and then relate them, leads to many of the benefits of a full data base management system.

Application generator programs can be called as subroutines from a COBOL program, and COBOL programs can be called from an application generator program. This is very important because of the number of already existing COBOL programs. The application generators' ability to select subsets of created files allows special purpose queries of your data base. This special purpose

query can be reported easily and flexibly using the non-procedural report commands.

Examples of Application Generators

Application generators are available for the small microcomputers (The Last One from Krown Computing) and increasingly large minicomputers (INFO from Henco, Inc.; IRMS from Xerox; FOCUS from Information Builders, Inc.; QUIZ and QUICK from Quasar Inc.; and RAPID 3000 from Hewlett Packard), as well as big computers (RAMIS from Mathematics Inc.; INQUIRE from Infodata Systems Inc.; MARK IV. from Informatics, Inc. and NOMAD from National CSS Inc.) (Couch, 1982; Crowder, 1981).

A CASE STUDY

College Background

The concerns described below led one college to employ an application generator, initially for institutional research applications. In early 1979, this college converted from an IBM360 batch system to a PRIME500 interactive, on-line system. There are now over forty terminals on-line with this PRIME computer. The PRIME's data files were not integrated; multiple independent (MIDAS) files were used. The priority in conversion was to have on-line student registration as soon as possible, and the college did not want this delayed to allow for an integrated

data base management system.

This college serves over 20,000 students each year generating almost 6,000 FTE (full-time equivalent) students. The student population is about 40% vocational, 40% lower division collegiate, and 20% community education and special programs (e.g., Developmental Education and English as a Second Language). As one would expect, administrative demands on the computer system have been heavy and growing. Following on-line student registration came a financial system (including payroll), and a personnel system. The college is currently bringing the COBOL-based financial and personnel systems on-line. Within this environment, it is understandable that programming in support of institutional research has been limited. Maintenance of existing systems is absorbing an increasing share of programming time. Without an integrated data base management system it has also been difficult to access the existing data, especially from different files. As with much of higher education, this college is also facing financial problems, due primarily to a depressed state economy. Thus, the college found itself with the wealth of computer based data needed for critical decision-making, but also faced with the problem of accessing and manipulating the data within an environment of increasingly scarce resources.

Institutional Research Background

Given the above environment the office of institutional research has not relied on programming support, but on the use of available, general purpose software packages. Initially, this included the PRIME's text editor and word processing systems (EDITOR and RUNOFF) and SPSS (Statistical Package for the Social Sciences) for statistical analysis. The advantages accruing to institutional research from on-line interactive computing and these types of software packages are almost universally known today. Quality control, standardization, relieving bottlenecks in report preparation, are just a few of the benefits.

Initital applications included data entry, analysis and report preparation for annual student follow-up studies and vocational program evaluation reports. SPSS is extensively used in preparing special one time studies on student retention and achievement as well as external reports such as the HEGIS enrollment reports. The only programming support required is COBOL generated data extracts from the student data base (MIDAS files).

Although this system led to improved productivity and quality control, there were several serious limitations. The text editor enhances data entry but does not allow for simultaneous data veri ication and editing, and is less efficient than input via specially designed screens. Further, relying on COBOL programming to access data is especially expensive when the data requested

resides in different files (e.g., the student demographic file and the student transcript file).

The Application Generator

These limitations led the institutional research office to investigate other software packages. This investigation resulted in the purchase of 'INFO', an application generator package developed by HENCO, Inc. of Wellesley, MA. INFO works on the basis of English language commands and allows the user to quickly create data processing systems. Some of INFO's major capabilities are described below:

1. User created screens for data entry, updating and viewing. This includes data editing and verification such as value ranges for data items and prevention of duplicate values being entered (i.e., SSN). Once the user has become familiar with INFO commands, sophisticated screens can be built in about an hour.

2. INFO also has the ability to access external MIDAS (ISAM) files and thus, eliminate the COBOL programming described above for data extracts.

3. Once data files have been built or extracted, up to ten files can be 'related' by any common data item. Once files are related, calculations can be performed on data items and

item values can be moved from one file to another.

4. Particular subsets of data can be selected from a file by specifying item values desired.

5. Reports can be easily generated with user defined titles and headings. Calculations can be performed with information like sub-totals, grand totals and averages.

6. All of the above capabilities can be placed within a user defined command stream using INFO programming. 'IF-THEN', 'GOTO', and other programming statements can also be incorporated into these command streams. Thus, an entire series of operations can be performed with one user defined command. INFO can also work in conjuction with other programs such as subroutines written in COBOL.

Institutional Research Applications

Within months of INFO's installment, most institutional research data processing had been converted to INFO. Significant improvements have been achieved in data entry, especially with regard to data collected by various surveys. Under the old system, data collected on questionnaires would have to be converted to code sheets for entry using the text editor. Now, the data can be entered directly from the questionnaires into specially designed screens. Once the data has been entered and

verified via INFO, it can be saved in an EDITOR file, accessed by SPSS for statistical analysis, and reported via our word processing system.

INFO's most significant impact for institutional research, however, has been the area of complying with various state and Federal reporting requirements. Often the data required existed in the computer but in different files, in different formats, and coded differently. Although a simple COBOL extract of data from one file is not time consuming, it would be if the program also had to pull together the data from different files, reformat, recode and verify. INFO's capabilities and flexibility has allowed this latter task to be accomplished efficiently and within a short period of time. Thus, INFO has resulted in less time being spent on government reporting while it has also increased the accuracy and confidence in these reports.

Other College Applications

In addition to meeting its own needs, the Office of Institutional Research has helped other areas use INFO to meet software needs which could not be met by Computer Services because of cost contraints and priorities. Because INFO is user-oriented and thus easy to learn, existing staff from other college offices were helped to develop their own data processing systems. These offices include the Business Office, Plant Services, Financial Aid Office, the Office of Instruction, and Admissions and Records.

Through these applications, INFO has paid for itself many times over in saved programming costs, improved productivity, and better service to students.

Major INFO systems developed to date include the college's budget development system, a student financial aid application system, a facility inventory system, and an equipment inventory system. Smaller INFO applications are constantly being developed to keep track of things like enrollment statistics, transcript requests, and instructor certifications.

The development and maintenance of these systems would not have been possible given existing finances without INFO programming capabilities. There are basically two types of programs used; 'menu' and 'processing' programs. As the name implies, 'menu' programs offer the user a set of choices. By entering the number corresponding to the choice desired, the user can quickly move through a series of operations in which file assignments and other command streams are transparent. These menu programs allow management and support staff to operate the system with little computer background or knowledge of INFO.

As an example, the main budget development system menu offers the user choices of adding or updating records, performing calculations, generating various reports, or branching to other menus (i.e., for updating salary schedules).

The 'processing' programs relate files by common items (e.g., SSN), sort files, perform calculations, and generate various reports. That is, they give the user the effectiveness and productivity that traditional programming offers. The major difference is that these programs are under user control and their maintenance requires only a fraction of the time needed for equivalent COBOL or other third generation programs. The reason for this is that INFO programs only require from 1/4 to 1/10 of the lines of code needed in an equivalent COBOL program. This leads to a very significant cost saving since over half of the cost during the lifetime of a COBOL generated system is in maintenance programming.

Further, because INFO is 'relational', the college can have many of the characteristics of a relational data base management system at a fraction of alternative costs. That is, the user can relate the different independent files to one another and pull together information that otherwise would not be available.

THE ROLE OF APPLICATION GENERATORS WITHIN INSTITUTIONAL RESEARCH AND THE INSTITUTION

The Role of Institutional Research

An institutional research office must transform collected data into the information needed for decision-making (Stevenson, 1981). The crucial problem of determining decision-makers' information

needs must be addressed first. Then, strategies for the transformation of data into the needed information must be designed. Since computers can quickly and reliably process data, their use must be considered in accomplishing the tasks of institutional research (McCorkle, 1977; Sandlin, 1977; Schelahan, 1977; Ryland, 1979 and Ridge, 1980). However, computer programming can simply either not be available, be too costly, or be too slow.

Importance of Application Generators to Institutional Research

The importance of application generators is that they allow the institutional research office staff to do the necessary programming at a reasonable cost. Further, application generators are flexibly designed to do exactly what the institutional research office must do; transform data into processed data which is hopefully relevant information. While data processors can process data efficiently, institutional researchers are specialists in keeping their eyes on the intended use of the processed data. Information for decision-making comes from non-routine, unique requests for data deliverable before a specified date. Although an application generator will not solve all problems, it will make an institutional research office more efficient, thus freeing up additional time. The application generator will allow the institutional research office to expand the range of information products that it can produce within existing resources and without expansive programming services.

As an institutional research office begins to provide more information, the demand for information will rise exponentially. To maintain a viable role, institutional research must be able to help their institution meet the increasing demand. This is a crucially important new role for institutional research. While institutional research will face an increasing demand for its services, someone must guide the institution and its other offices toward more efficient use of computing resources within developing the information necessary for decision-making.

Why Institutional Research, and Not Data Processing Should Be The Catalyst

Institutional research should be the catalyst for this change for three reasons. First, if the intended result is 'information', then the information development specialist of a research office should be involved rather than the 'data' processors (Kriegbaum, 1980 and Strenglein, 1980). Second, application generators represent a significant change over present data processing techniques; thus, there is a natural resistence to change by the data processors who have traditionally held tight controls over all software systems (Bronstein, 1982). Institutional research's focus on information development makes it the best suited change agent. Third, since data processing is already overloaded and probably not spending the time it should on the strategic isues of coordination, security, control, consistent operations, data base

management, and data communications, it is entirely appropriate for another office, the institutional research office, to guide the non-data processing college staff into the world of fourth generation software such as application generators.

Institutional Benefits

The bottom line is what is best for the institution. Institutional research offices have a unique opportunity to play a strategic role within developing their institution's information systems. Application generators are productivity tools which are especially critical during times of shrinking resources (Codkind, 1982). Application generators can save a great deal of money by allowing staff in all college offices to do more.

Application generators will allow the potential of the ever improving computer hardware to be tapped. The importance of this can be seen in that computers and computing are becoming simpler. Yet, because of their technical complexity, computers necessarily have been defended with a verbal arsenal called 'computerese.' Any computer user intrepid enough to approach the computer fortress with 'A REQUEST' has sometimes been dispatched from this world of boolean logic with a barrage of bits, bytes, and binary babble. After this, the brave user too often departed this kingdom with less enthusiasm to return.

This type of communication barrier is a major concern confronting

computing. The computer users' lack of computer literacy and the data processors' only rudimentary understanding of unique business circumstances has become a major bottleneck. By 1990, the U. S. Department Of Labor has 'estimated that 50% of the U. S. workforce will depend on some form of information processing and 20% of the labor force will be in it!' (Smith, 1982).

Having technology and using it are two different things. The idea of a 'Management Information System' that supplies decision-making information is a dominant goal for which we all reach. If present trends continue, the already foreseeable improvement in computer hardware will make this MIS goal dependent primarily on software development.

Without question, the most difficult challenge willl be to provide adequate software for the spiraling number of users. There are a variety of ways to improve the productivity of programmers. However, these methodologies and improvement techniques will not satisfy the demand for new applications. The number of programmers produced per year is not growing near the rate needed to supply the software needs of business.

A similar situation occurred in 1911 when a prediction was made ... given the accelerating use of telephones compared to U. S. population, that by 1939 every American woman under the age of 16 would need to be a telephone operator. The irony is, the prediction underestimated the growth of telephones. By 1939 every

man, woman, and child in America was a telephone operator ... because technology allowed them to be. Telephone users could do it themselves for most situatons, while operators now devote their time to specialized situations (Smith, 1982).

Something similar must happen if the goals within the capabilities of the computer's hardware are to be realized. Information systems will need to be 'user friendly' ... that is, English speaking, business-oriented, simple-to-use, and most importantly - user programmable.

Meanwhile, professional programmers will need to focus on the difficult, unique, special jobs that are beyond the capabilities of user programming. Computer users will need not only to become computer literate, but also to learn to use a new-generation software. Further, all managers will need to be involved, at least in the software decisions.

The emerging new generation of software has the potential to unlock the traditional mystique of the computer from its boolean barrage of computerese. The use of general purpose software such as application generators, is a first step in these directions.

The use of application generators will allow the educational institution to be a leader and for it to save much needed time and money.

STRATEGIC ISSUES

Getting Started

An application generator is a tool which is productive only after the data processing environment and users are ready. Thus, the following are suggested positions for the institutuional research office before introducing the application generator ...

1. Support the acquisition and implementation of an on-line, interactive computer system.
2. Support the development of a data base management plan by data processing and preferably with your help.
3. Obtain on-line interactive access to the computing resource and the data base.
4. Systematically get research office people to learn the basics of the computer's operating system and it's editor.
5. Obtain a statistical processing and a word processing system which works within your computer's operating system and editor.
6. Spend at least six months and preferably over a year making profitable use of these on-line systems and learning how to get at the data you need from the computer.
7. Investigate application generators which are available for your computer system and which will solve your problems.

As previously mentioned, application generator languages are becoming available on MICROCOMPUTERS which may significantly

change the above sequence of positions.

After the application generator is obtained, the following strategies are suggested ...

1. Above all ... go slow, this is a new and slightly different way to utilize the computer. Do not expect much help, especially from campus data processors ... however, court their involvement.
2. At the beginning identify one or two other key computer users who have identified projects which are far down the priority list for programming services. Convince these users to be trained as your staff is trained.
3. Take a leadership position during this training. Conduct weekly sessions on the use of the application generator software.
4. Develop one or two prototype systems while keeping track of time and other resources used.
5. Present to higher management a demonstration of the developed system and point out the many possible institutional benefits.

Where Will This Go?

While the future is difficult to forecast, the following strategic issues may arise ...

1. The role of the institutional research office may shift toward helping users develop their own information systems.

2. The application generator probably will lead to much less and possibly no further COBOL based development.

3. The college will need to articulate a relationship between the COBOL-type, third generation software development and the application generator-type fourth generation software. As users on campus start developing their own systems, new processes will have to be developed 1) to inform data processing about emerging new systems; 2) to certify a new system is ready for operation with a contract identifying who is responsible for documentation, system maintenace, and system operation; 3) to control data base access and security; and 4) to prioritize access to computer resources.

4. The user freedom to develop possible systems without undue controls will be at odds with the traditional, highly controlled COBOL-type system development process.

5. Data processing's focus will move from application system development toward 1) control and security of computer usage, 2) data base management, 3) distributed processing, and 4) data communication.

6. The longer run issue will be what capabilities to provide in the automated office of the future. The offices of the not-to-distant future may use their computer terminals for 1) electronic mail, 2) word processing, 3) data base access,and of course, 4) application system development.

CONCLUSION

The paper has described the importance of application generator software to one college's institutional research office, as well as to the process for developing information systems. Literature, as well as this college's experience, documents that application generators can play a crucial and strategic role within institutional research and probably for the total institution.

REFERENCES

Bradley, J., 'The Elusive Relation' Computerworld. In-Depth section, March, 1982.

Bronstein, P., 'Development Software Helps Users Program' Small Systems World. Volumn 10, number 5, 1982.

Codkind, A., 'For General Application's Development, Go ' NON-PROCEDURAL' to Close Productivity Gap' Computerworld. Special Report, April 1982.

Couch, J., 'Datagramming', Apple Orchard. Volumn 2, no. 4, Winter, 1982.

Dalton, R., 'Are Programmers Really Necessary...A Survival Guide to User Application Development For MIS Managers.' Schindler. 1981.

Dodge, J., 'Automators Will Be Leaders-Martin.' Software News. December 7 1981.

Dodge, J., 'Structured Programs Out User Programs In - Martin.' Software News. December 7 1981.

Hoffman, R., '1000% Boost in Software Productivity Offered As

Target.' Information System News. Monday, May 4 1981.

Kriegbaum, R., 'Integrating the Institutional Decision Process' Cause/Effect. Volume 3, number 5, pp28-32, September 1980.

Lerner, N., Brownstein I., and Smith W., 'Winds of Change: The Impart of Fourth-Generation Languages on Documentation.' Computerworld. 1982.

Morton, S., Management Decision Systems. Boston. Harvard University Graduate School of Business Administration.

McCorkle, C., Jr., 'Information For Institutional Decision Making.' New Directions For Institutional Research. No. 15, autumn, 1977.

McCrackin, D., 'The Changing Face of Applications Programming.' Datamation. November 15, 1978.

Ridge, J., 'The Office Minicomputer: The Potential for Institutional Research' AIR Forum Paper. May 1980.

Rochester, J., 'Daring Attitude Required in DP Productivity - Martin.' Computerworld. September 28, 1981.

Ryland, J., 'Proprietary Software and The Institutional Researcher.' New Directions For Institutional Research. No. 22,

1979.

Sandin, R., 'Information Systems and Educational Judgement'. New Directions For Institutional Research. No. 15, autumn, 1977.

Schindler, P., 'Study Sees Application Programmers Extinct.' Information system News. January 12, 1981.

Shelhan, B., 'Reflections on The Effectiveness of Informational Support For Decision makers.' New Directions for Institutional Research. No. 15, autumn, 1977.

Smith, C., 'Software Performance Engineering' Computerworld. In-Depth section, March 1982.

Snyders, J., 'Those Belated Distributed DBMSs.' Computer Decisions. February 1981.

Staman, E., 'Computing and Office Automation - Changing Variables.' The AIR Professional File. No. 10 (Summer/Fall) 1981, pp. 1-6.

Staman, E., 'Trends in Adminstrative Computing.' New Directions For Institutional Research. No. 22, 1979.

Strenglein, D., 'Information System Management: A Function for Institutional Research' AIR Forum Paper. May, 1980.

Stevenson, M., 'Proposed Self-Assessment Guide for Data and Information Processing Activities' Oregon Department of Education. Paper, April 1981.

Thomas, C., 'Nonproprietary Software As An Institutional Resource.' New Directions For Institutional Research. No. 22, 1979.

Watkins, G., 'What Are Application Generators.' Computerword. November, 1980.